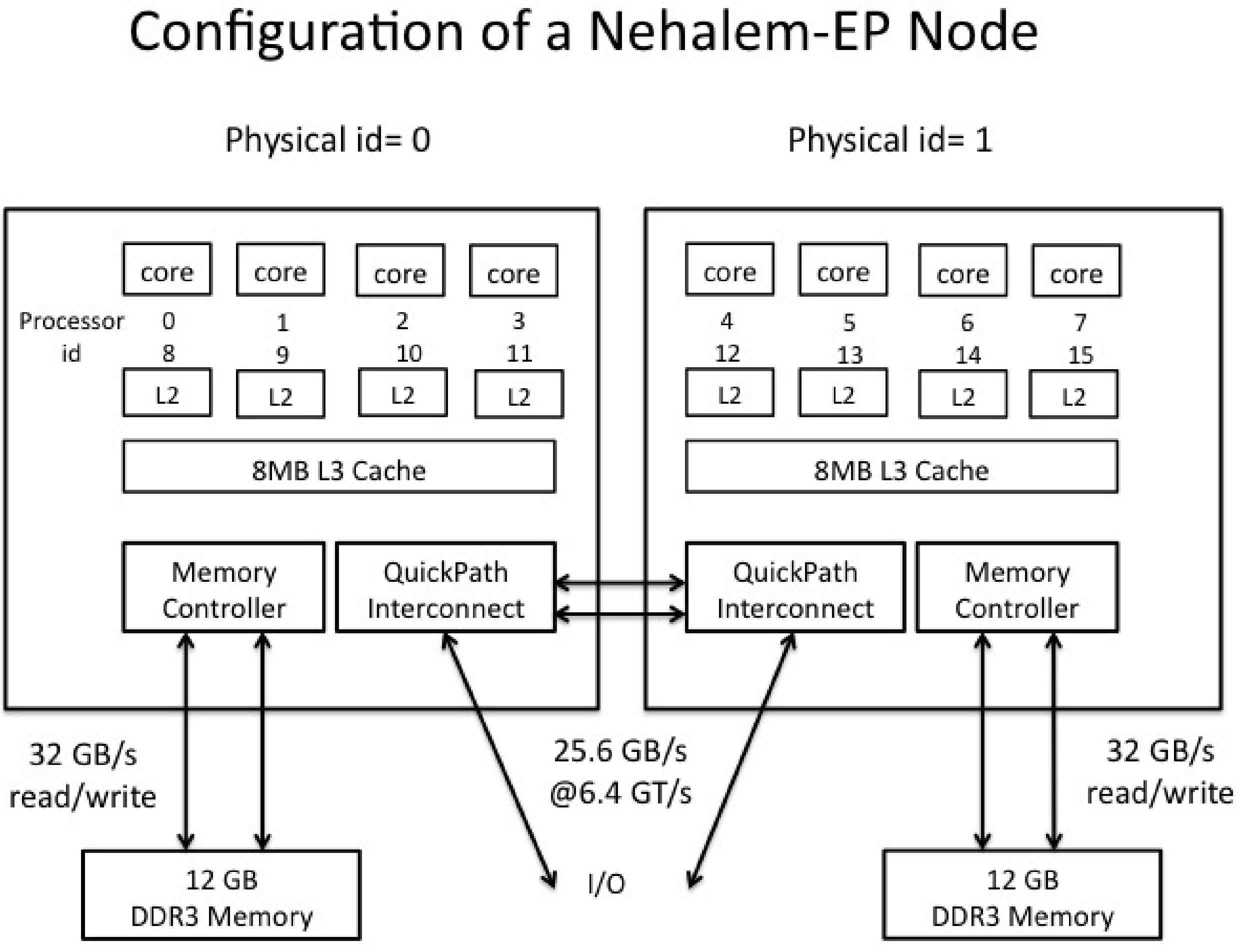
Nehalem-EP Processors

Category: Pleiades

DRAFT

This article is being reviewed for completeness and technical accuracy.

Configuration of a Nehalem-EP node:



Core Labeling:

Unlike Harpertown, the core labeling in Nehalem-EP (and also Westmere) is contiguous. That is, cores 0-3 are in first socket and cores 4-7 are in the second socket.

When using the SGI MPT library, the enviroment variable **MPI_DSM_DISTRIBUTE** is set to *on* by default for the Nehalem-EP (and also Westmere) nodes.

SSE4 Instruction Set:

Intel's Streaming SIMD Extensions 4.2 (SSE4.2) instruction set is included in the Nehalem-EP processors.

Since the instruction set is upward compatible, an application that is compiled with -xSSE4.1 (with Intel version 11 compiler) can run on either Harpertown or Nehalem-EP or Westmere processors. An application that is compiled with -xSSE4.2 can run ONLY on Nehalem-EP or Westmere processors.

If you wish to have a single executable that will run on any of the three Pleiades processor types with suitable optimization to be determined at run time, you can compile your application with -O3 -ipo -axSSE4.2,xSSE4.1

Hyperthreading:

On Nehalem-EP (and also Westmere), hyperthreading is available by user request, for example by asking for more than 8 MPI ranks per Nehalem-EP node.

When hyperthreading is requested, the OS views each physical core as two logical processors and can assign two threads to it.

Preliminary benchmarking by NAS shows that many jobs would benefit from using hyperthreading. Therefore, it is currently turned ON, meaning that it is available if a job requests it.

Mapping of Physical Cores and Logical Processor IDs

Physical id	**Core id**	**Processor id** Hyperthreading **OFF**	**Processor id** Hyperthreading **ON**
0	0	0	0 ; 8
0	1	1	1 ; 9
0	2	2	2 ; 10
0	3	3	3 ; 11
1	4	4	4 ; 12
1	5	5	5 ; 13
1	6	6	6 ; 14
1	7	7	7 ; 15

With hyperthreading, one can run an MPI code with 16 processes instead of just 8 per

Nehalem-EP node. Each of the 16 processes will be assigned to run on one logical processor. In reality, two processes are running on the same physical core. If one process does not keep the functional units in the core busy all the time and can share the resources in the core with another process, then running in this mode will take less than 2 times the walltime compared to running only 1 process on the core. This can improve the overall throughput as demonstrated in the following example:

Example: Consider the following scenario with a job that uses 16 MPI ranks. Without hyperthreading we would use:

```
#PBS -lselect=2:ncpus=8:mpiprocs=8 -lplace=scatter:excl
```

and the job will use 2 nodes with 8 processes per node. Suppose that the job takes 1000 seconds when run this way. If we run the job with hyperthreading, e.g.:

```
#PBS -lselect=1:ncpus=16:mpiprocs=16 -lplace=scatter:excl
```

then the job will use 1 node with all 16 processes running on that node. Suppose this job takes 1800 seconds to complete.

Without hyperthreading, we used 2 nodes for 1000 seconds (a total of 2000 node-seconds); with hyperthreading we used 1 node for 1800 seconds (1800 node-seconds). Thus, under these circumstances, if you were interested in getting the best wall-clock time performance for a single job, you would use two nodes without hyperthreading. However, if you were interested in minimizing resource usage, especially with multiple jobs running simultaneously, use of hyperthreading would save you 10%.

An added benefit of using fewer nodes with hyperthreading, is that when Pleiades is loaded with many jobs, asking for half as many nodes may allow your job to start running sooner, resulting with an improvement in the throughput of your jobs.

Caution: Hyperthreading does not benefit all applications. Some applications may also show improvement with some process counts but not with other process counts (e.g., a 256-process Overflow job shows benefit with hyperthreading, while a 32-process Overflow job does not). There may also be other unforeseen issues with hyperthreading. Users should test their applications with and without hyperthreading before making a choice for production runs. If your application runs more than 2 times slower with hyperthreading than without hyperthreading, then it should not be used.

Turbo Boost:

On Nehalem-EP (and also Westmere), Turbo Boost is available.

When Turbo Boost is enabled, idle cores are turned off and and power is channeled to the cores that are active, making them more efficient. The net effect is that the active cores perform above their clock speed (i.e., overclocked).

Turbo Boost mode is set up in the system BIOS. It is currently set to ON.

Article ID: 79
Last updated: 27 Oct, 2011
Computing at NAS -> Computing Hardware -> Pleiades -> Nehalem-EP Processors
http://www.nas.nasa.gov/hecc/support/kb/entry/79/?ajax=1